(19) Europäisches Patentamt
European Patent Office
Office européen des brevets

(11) Publication number: **0 079 995**
**A1**

# (12) EUROPEAN PATENT APPLICATION

(21) Application number: 81305553.0

(22) Date of filing: 24.11.81

(51) Int. Cl.³: G 06 F 9/32

(43) Date of publication of application: 01.06.83 Bulletin 83/22

(84) Designated Contracting States: DE FR GB

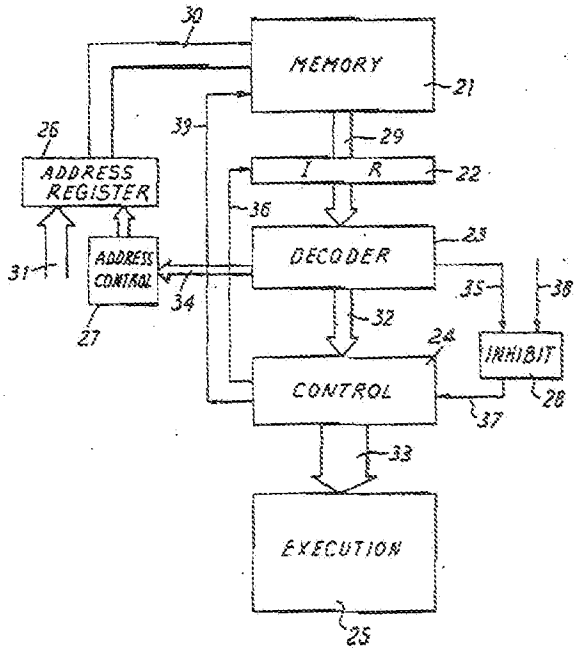
(71) Applicant: NIPPON ELECTRIC CO., LTD., 33-1, Shiba Gochome, Minato-ku, Tokyo, 108 (JP)

(72) Inventor: Kimoto, Masabu, c/o Nippon Electric Co, Ltd. 33-1, Shiba Gochome, Minato-ku Tokyo (JP)

(74) Representative: Pears, David Ashley et al, REDDIE & GROSE 16 Theobalds Road, London WC1X 8PL (GB)

## (54) Information handling apparatus having an instruction-executing function at a high speed.

(57) A memory (21) stores instructions which are addressed by an address register (26) and read into an instruction register (22). A decoder (23) decodes each instruction in the register and effects the necessary control operations via a control unit (24). The control unit in turn controls the execution logic (25) whereby the data processing operation required by the instruction is performed. A circuit (28) detects when the processing required by the instruction is to be inhibited. Address control means (27) advance the instruction address in the register 26 and, when inhibiting takes place, the means (27) effect the advance in a shorter time than that allotted to the inhibited processing, in order to speed up the average rate of execution.




EP 0 079 995 A1

# INFORMATION HANDLING APPARATUS HAVING AN INSTRUCTION-EXECUTING FUNCTION AT A HIGH SPEED

The present invention relates to an information handling apparatus, and more particularly to improvement in an apparatus comprising means for reading memory contents, means for decoding the read memory contents and means for executing a processing on the basis of the decoded results.

An operation of the respective means of the mentioned apparatus is controlled by at least one timing signal produced by timing control circuit, and thereby a desired program is executed. Such apparatus has been heretofore generally used as a principal section, i.e., a program processing section of every information handling instrument including a large-scale computer to a micro-computer. This apparatus normally comprises a memory for storing a group of instructions (commands, subroutines, microprograms, etc.) which are necessitated for executing a program, in a binary coded form, a circuit for designating a particular instruction in the memory (an addressing circuit), a circuit for reading the designated instruction from the memory, a circuit for temporarily holding the read instruction (an instruction register), a circuit for decoding the read instruction (a decoder), a circuit for generating at least one control signal on the basis of the decoded result (a timing control circuit) and a circuit responsive to the control signal for executing the processing determined by the instruction (an execution

circuit). The addressing circuit includes an address register, whose content represents an address for the memory. Accordingly, by sequentially modifying the content of the address register, instructions necessitated for executing a given program are successively read out of the memory into the instruction register. This instruction designates an operation of the execution circuit, and there are many varieties of instructions such as, for example, calculation instructions, data transfer instructions, a subroutine call instruction, a jump instruction, etc. It is to be noted that an instruction has to be executed within a predetermined period. This execution period is defined by at least one cycle time of a machine (normally called "machine cycle"). Furthermore, the execution period of an instruction is not always constant for every instruction. In other words, a number of necessary machine cycles is determined depending upon the instruction. For instance, for an instruction of merely adding a content of an A-register to a content of a B-register, only one machine cycle suffices. However, an addition instruction of adding a content of a memory location whose address is designated by a content of an HL-register to a content of an A-register, necessitates two machine cycles. Moreover, some of complexed instructions necessitate 5 machine cycles or 6 machine cycles. Accordingly, a sum of the machine cycles allotted to the respective instructions, is a processing time of the program.

On the other hand, a number of bits of an instruction that can be read out of a memory by one access is equal to a number of bits of a data bus. That is, if a data bus consists of a one-bit line, then an instruction code of one bit can be read. And if a data bus consists of a 4-bit line, then an instruction code of 4 bits can be read out in parallel in the same timing. Normally, one instruction is coded with a plurality of bits, and in order to achieve speed-up of a processing, design is made such that an instruction code of a plurality of bits may be read out in parallel. For instance, 4 bits, 8 bits, 16 bits, etc. are simultaneously read out of a memory as a unit (one byte). Further, if each one instruction is limited to one byte, then a number of available instructions would be limited. For example, if one byte consists of 4 bits, then the number of the available instructions is $2^4 = 16$. In order to increase the number of the available instructions, it is only necessary to increase the number of bits per one byte, but in that case a number of wirings for a data bus or an address bus is increased, and so this is not favorable, especially in a small-sized type of machines. Therefore, in the prior art, it was contemplated to overcome this shortcoming by increasing a number of bytes per instruction. In this case, as a matter of course, a number of accesses to a memory is increased. For instance, if one instruction consists of 3 bytes, then three times of memory access are necessitated. To that end, an addressing circuit must perform a processing of modifying a content of an address register three times. Normally the respective bytes forming one instruction

are allotted in consecutive address spaces. Accordingly, the addressing circuit has a facility of sequentially adding one to an address of a leading byte of an instruction (increment facility). However, since at least one machine cycle is generally necessitated for one memory access, for instance in order to read a 3-byte instruction at least 3 machine cycles are necessitated, and furthermore, additional machine cycles for processing the read instruction is also necessitated. Therefore, the fact that if a number of bytes in an instruction is increased, then a processing speed is correspondingly lowered, was an inevitable shortcoming in the prior art.

On the other hand, in order to achieve speed-up of a processing it may be conceived to increase a frequency of a clock signal for controlling the minimum operation timing of an apparatus. However, integrated circuit elements which can follow such a high-speed clock signal have not been mass-produced so far. Especially, the MOS transistors adapted to integration circuit are disadvantageous in that a clock signal with a high frequency can not use. Even if integrated circuit elements operable in response to the high-frequency clock signal should be mass-produced, it would be naturally expected that a further high-speed program processing is desired.

On the other hand, as would be understood from checking execution steps of many programs, all those instructions, which are read out of a memory by address designation, are not always to be executed. Because, it is difficult for a programmer to control a sequence of

memory accesses corresponding to every programming condition. Therefore, the program sometimes contains instructions which are not to be executed, although they are read out of the memory by the memory access.

Further, there is frequently such an occasion that a programmer makes a program so as to prohibit the execution of an accessed instruction under a predetermined condition. In this occasion, the execution of the instruction is prohibited by another instruction or by a control signal from a condition detect circuit working independently of instructions. Of course, there are some cases where a condition of the apparatus is judged by an instruction and execution of the instruction which is read out of the memory is prohibited according to the judged condition.

In the prior art, the instruction, the execution of which should be cancelled or inhibited, has to be exchanged to a non-operation (NOP) instruction. This NOP instruction is an instruction that keeps a condition in an apparatus as it is. However, an execution period of the NOP instruction has to be equal to an essential execution period allotted to the instruction for the inhibited execution. For instance, if an instruction, which should be inhibited, is a 2-machine cycle instruction, the NOP instruction necessitates two machine cycles. If a canceled instruction is a 5-machine cycle-instruction, the NOP instruction needs five machine cycles in its processing.

Although this NOP instruction is necessary in a program execution, a time period for its execution should be as short as possible. However, in the prior art, the time period of the NOP instruction can not be shortened because of necessity of memory access.

It is therefore one object of the present invention to provide an information handling apparatus which can execute a high-speed program processing.

Another object of the present invention is to provide an information handling apparatus having a novel processing facility such that after a particular instruction, without executing at least one scheduled instruction the processing can be shifted to execution of another instruction at a high speed.

Still another object of the present invention is to provide an information handling apparatus in which preparation of discrete or discontinuous addresses for memory access can be performed at a high speed.

Yet another object of the present invention is to provide an information handling apparatus in which a time period till a scheduled instruction is shifted without its execution can be shortened.

A further object of the present invention is to provide an information handling apparatus having a novel circuit with an address modification.

A still further object of the present invention is to provide an information handling apparatus having a novel facility such that for

an instruction consisting of a plurality of bytes, without accessing to memory locations where instruction codes of the second and subsequent bytes are stored the next new instruction can be accessed.

A yet further object of the present invention is to provide an information handling apparatus including a circuit which can be achieve a processing that among instructions subsequent to a particular instruction, an instruction of the same type as the previously executed particular instruction is omitted from execution at a high-speed.

A still further object of the present invention is to provide an information handling apparatus including a control circuit which enables to skip a scheduled instruction and read out the next instruction in a period that is shorter than the period necessitated for executing the scheduled instruction.

The information handling apparatus according to the present invention comprises a memory in which a group of instructions are stored, first means for designating a memory location where the instruction to be read is stored, second means for reading the designated instruction out of the memory, third means for detecting that a predetermined instruction has been read out and generating a detection signal, forth means for inhibiting execution of at least one subsequently scheduled instruction in response to the detection signal, fifth means for preparing in a short period of time an address for a next new instruction of the instruction inhibited from execution, and sixth means for executing the new instruction read out in response to

the prepared address. "An instruction whose execution is inhibited" as defined according to the present invention means an instruction regulated so that although at least one portion of the instruction or an information with respect to the instruction is read out of the memory by memory accessing, the processing defined by the instruction is not executed. In other words, it means such instructions that the results obtained by executing the instruction should be disregarded, or that at the time point there is no need to process as defined by the instruction, or that the processing defined by the instruction should be omitted under a predetermined condition. Typically, one of such instructions are an instruction subsequent to a skip instruction, instructions subsequent to a particular instruction in an instruction group consisting of a plurality of consecutive instructions of the same type, and the like, as will be fully described later.

These instructions are different from the conventional branch instruction. Namely, they are such instructions that in the prior art they were once read out in response to memory access but under a certain condition their inherent processings were omitted without being executed. The location in a memory where such instruction is physically set, could be either just subsequent to the instruction defining the certain condition or remote from the latter instruction. According to the present invention, in order to shorten a processing time for such instruction, there is provided circuit means for preparing an address of the next instruction at a high speed. In other words,

the information handling apparatus according to the present invention comprises means for replacing an address of an instruction not to be executed by an address of the next instruction to be executed. Accordingly, the time required when such an instruction is detected and the omitted can be made very short, and hence sequence control of a program can be executed at a high speed. Consequently, even though an operation clock frequency is not raised, a program processing period can be sufficiently shortened, upon applying the present invention to an apparatus employing the heretofore known integrated circuit elements operable only at a relatively low speed. In addition, as will be described later, the present invention is especially effective in the case of canceling execution of an instruction consisting of a plurality of bytes. Moreover, the present invention is also effective for an instruction which necessitates a plurality of machine cycles, even when the instruction is a one-byte instruction.

The above-mentioned and other objects, features and advantages of the present invention will become more apparent by reference to the following description of preferred embodiments of the invention taken in conjunction with the accompanying drawings, wherein:

Fig. 1 is a block diagram of an information handling apparatus in the prior art;

Fig. 2 is an operation timing chart for the prior art apparatus illustrated in Fig. 1;

Fig. 3 is a block diagram illustrating one preferred embodiment of the information handling apparatus according to the present invention;

Fig. 4 is a block diagram showing one practical example of an address control circuit in the apparatus illustrated in Fig. 3;

Fig. 5 is a block diagram showing another practical example of the address control circuit;

Fig. 6 is a block diagram illustrating an essential part of another preferred embodiment of the information handling apparatus according to the present invention;

Fig. 7A and 7B, respectively, shows an operation timing chart of the prior art apparatus and an operation timing chart of the apparatus shown in Fig. 6; and

Fig. 8 is a block diagram illustrating an essential part of still another preferred embodiment of the present invention.

Referring now to Fig. 1, an information handling apparatus in the prior art is illustrated as a block diagram. A group of instructions which are used to program processing are preliminarily stored in a memory 1, in which a necessary instruction is designated by a content (an address) of an address register 6. To the memory 1 are coupled an address bus 10 and a data bus 9. These buses are respectively formed of a plurality of signal lines so as to be able to handle a plurality of bits simultaneously for executing a high-speed processing. An instruction designated by the content of the address register 6 is read out the memory 1 by a read-out control signal 18 from control

circuit 4. Then, an instruction code in the first byte thereof is fetched in an instruction register 2 in response to an instruction fetch signal 16 issued from a control circuit 4. This instruction code is decoded by a decoder 3, and the results are sent to the control circuit 4 via a bus 12. The control circuit 4 produces various kinds of control signals 13 and transmits them to an execution circuit 5 for processing a program, on the basis of the decoded results. The execution circuit 5 comprises element means used for executing a program such as registers, an arithmetic-logic unit, gate circuits, etc., and they execute a read-out instruction as controlled in timing by the control signals 13. The control circuit 4 applies an address advance signal 14 to an incrementer 7 in order to read out an instruction to be subsequently executed. The incrementer 7 has an addition facility for incrementing the present content in the address register 6 by +1 in response to the address advance signal 14. Then, the result of the addition (the next address) is set in the address register 6. It is to be noted that the address register 6 is programmable. Namely, any desired address can be set therein via a bus 11 and its content can be reset. In the case where one instruction is formed of a plurality of bytes, it is necessary to designate as many addresses as the bytes. For instance, in the event that one instruction is formed of 3 bytes (3-byte instruction), three addresses generally represented n, n+1 and n+2 can be produced. These addresses are prepared by the incrementer 7. Since at least one machine cycle is required for once

of memory access, for the purpose of reading out a 3-byte instruction, at least 3 machine cycles are necessitated. In practice, since a time period for executing a processing on the basis of the instruction is also necessary, 4~6 machine cycles are required.

When a certain instruction is read out of the memory 1 and set in the instruction register 2, this instruction is decoded by the decoder 3. Now, if a condition for canceling the instruction is fulfilled in the apparatus, a control signal 15 is generated to activate an inhibit circuit 8. As a result, an inhibit signal 17 for inhibiting execution of that instruction is sent to the control circuit 4. This inhibit signal 17 serves to inhibit output of the control signals 13 which are normally generated on the basis of the decoded result. Accordingly, the execution circuit 5, which is not applied with the control signals 13, would not execute the inherent processing defined by the instruction in question, but would be held in the previous condition. In the prior art, this time period for holding the previous condition was equal to the time period necessitated for executing the inherent processing defined by the instruction.

The mode of the above-described operations will be now explained with reference to the timing chart in Fig. 2.

In Fig. 2, reference symbols $M_1$ ~ $M_8$, respectively, denote successive machine cycles. While the machine cycles are normally allotted independently for the respective instructions, here for convenience of explanation, consecutive suffixes are added to the symbols

M representing successive machine cycles. That is, description will be made here on instructions to be executed in the 8 machine cycles $M_1$ to $M_8$. Depending upon an apparatus, the durations of machine cycles are different, but here it is assumed that all the machine cycles have the same duration. However it is to be noted that the present invention is well applicable to an apparatus having machine cycles of different durations. In Fig. 2, A, B, C, D and E denote respective instructions, and parenthesized symbols n, n+1, n+2, n+3 and n+4 noted thereunder means addresses for designating the corresponding instruction. Furthermore, consecutive symbols $B_1$ and $B_2$ indicate that the instruction B is a 2-byte instruction, and they denote the instruction codes of the first byte and the second byte, respectively.

Fig. 2 (i) shows the mode of operation in which for a 1-byte instruction A, a 2-byte instruction B and a 1-byte instruction C, processings inherently defined by the respective instructions are executed, and Fig. 2 (i)' shows the mode of operation in which among the above-mentioned instructions, for the 2-byte instruction B the processing is made to be not executed. Fig. 2 (ii) shows the mode of operation in which the respective instructions are to be normally executed, in the case where all the instructions A~E are 1-byte instructions and among these instructions the instruction C and the instruction D, respectively, necessitate 2 machine cycles and 3 machine cycles, and Fig. 2 (ii)' shows the mode of operation in the case where among the above-mentioned instructions the consecutive instructions C

and D are made not to be executed.

In the following, the above-referred respective modes of operations will be described in more detail with reference to Fig. 1.

In the mode of operation shown in Fig. 2 (i), at first, an address (n) is set in the address register 6, and thereby the instruction A is read out of the memory 1. The read instruction A is a 1-byte instruction, which is decoded by the decoder 3, and the decoded result is transmitted to the control circuit 4. On the basis of this decoded result, the control circuit 4 generates control signals 13 which are necessitated for execution of the instruction A, and in response to the control signals 13 processing is executed in the execution circuit 5. It is assumed that this processing can be executed in one machine cycle $M_1$. When the execution of the instruction A has been finished (practically in the last state in the first machine cycle $M_1$), the control circuit 4 generates an address advance signal 14 for incrementing the content of the address register 6 by +1. As a result, the present content (n) of the address resister 6 is modified to (n+1) by the incremeter 7. The next machine cycle $M_2$ is a cycle allotted to the instruction B, and since the instruction B is a 2-byte instruction, in the machine cycle $M_2$ is read the instruction code $B_1$ of the first byte of the 2 bytes according to the modified address (n+1). In this instruction code $B_1$ is set an information indicating that the instruction B is a 2-byte instruction. Accordingly, that information is decoded by the decoder 3, and hence a procedure for reading the next subsequent

instruction code $B_2$ is effected. This procedure includes the processing for modifying the content of the address register 6 from (n+1) to (n+2) by controlling the incrementer 7. And in the next machine cycle $M_3$ is read the instruction code $B_2$ of the second byte. It is assumed that this instruction B necessitates 5 machine cycles in total before the processing designated by the instruction has been completely executed. Accordingly, the processing for the instruction B can be terminated at the end of the machine cycle $M_6$. It is to be noted that during the machine cycles $M_4$ to $M_6$, the content (n+2) of the address register 6 is not modified. This is because during these machine cycles the address advance signal 14 for controlling the incrementer 7 is not generated by the control circuit 4. A next address advance signal 14 is generated in a last state of the machine cycle $M_6$, thereby the content of the address register 6 is modified from (n+2) to (n+3), and so, in the next machine cycle $M_7$ the 1-byte instruction C is read out of the memory 1 and executed. In this way, the instructions A, B and C are successively executed in the seven machine cycles $M_1$ to $M_7$.

Whereas, in the case where among these instructions A, B and C, the instruction B is not necessary to be executed, the mode of operation will take the form illustrated in Fig. 2 (i)'. More particularly, the two instruction codes $B_1$ and $B_2$ forming the 2-byte instruction B are read in the instruction register 3 during the machine cycles $M_2$ and $M_3$, respectively, and they are decoded by the decoder 3. However, since a signal 15 indicating non-execution is generated, the inhibit circuit 8

is actuated to transmit an inhibit signal 17 to the control circuit 4, so that the control signals 13 in response to the instruction B may not be transmitted from the control circuit 4 to the execution circuit 5. Accordingly, during the period consisting of the machine cycles $M_2$ to $M_6$ which are normally necessitated for executing the instruction B, the execution circuit 5 makes no operation. In other words, a wasteful time is spent in the machine cycles $M_2$ to $M_6$.

In the mode of operation illustrated in Fig. 2 (ii), the instructions A, B, C, D and E, which are all one byte instructions, are executed, and among these instructions, the instructions C and D respectively necessitate 2 machine cycles and 3 machine cycles as a read-out and an execution periods. Accordingly, in the case where these instructions C and D are respectively inhibited from execution by the inhibit circuit 8 (Fig. 2 (ii)'), the execution circuit 5 also wastefully spends a time without executing anything during the machine cycles $M_3$ to $M_6$.

As described above, in the heretofore known information handling apparatus, in the case where there occurs an instruction that is not to be executed in a routine which must be passed in view of convenience for a program processing, the time period which is inherently necessary for executing the instruction was elapsed under a non-operation condition. Moreover, this time period is not a period reserved for controlling an operation timing in the apparatus, but reserved merely for the purpose of waiting the next new instruction without executing the intended instruction. Accordingly, it is desirable

to make the period under the non-operation condition as short as possible. However, in the prior art apparatus, since a facility of shifting from the instruction to a next new instruction at a high speed was not provided, it was impossible to shorten the period of the non-operation condition. One of the reasons is that addressing means for designating a next instruction is uniquely fixed so as to operate in response to machine cycles necessary to executed the present instruction. Moreover, another reason is that the apparatus is designed in such a manner that when the instruction not to be executed is formed of a plurality of bytes, all the bytes have to be read out of memory before shift to a next new instruction. Accordingly, the processing speed of the heretofore known apparatuses was, in any event, determined solely by the frequency of the operation clock applied to the apparatus and the procedure of processing a program, and it was impossible to further increase the processing speed.

An information handling apparatus according to one preferred embodiment of the present invention is illustrated in a block diagram of Fig. 3. A group of instructions, which are used for executing a program, are stored in a memory 21, and each memory location corresponding to instructions therein is designated by a content (address) of an address register 26. Then a disgrated instruction is read out of memory 21 in response to a read-out control signal 39. The read-out instruction is fetched into an instruction register 22. The address is transferred in a parallel form of a plurality of bits (for instance, 8 bits)

through an address bus, and the instruction is read out in parallel form of a plurality of bits (for instance, 4 bits) through data bus 29. A fetch operation to the instruction register 22 is carried out in response to a fetch control signal 36 is sued from a control circuit 24. The instruction set in the instruction register 22 is decoded by a decoder 23, and signals 32 representing the results of decoding are transferred to the control circuit 24. The control circuit 24 applies control signals 33, which are necessitated for executing the instruction, to a program execution circuit 25 on the basis of the decoded results. The execution circuit 25 executes the read-out instruction in response to the control signals 33. The above-mentioned process is similar to that in the conventional information handling apparatus in the prior art as described previously with reference to Figs. 1 and 2. Further, this embodiment has a novel address control circuit 27.

Now let us consider the case where an instruction where execution should be canceled, is handled. In other words, it is assumed that for an instruction, which should be accessed, there occurs a condition that it is unnecessary to execute this instruction. In this case, a signal 35 indicating that execution of the instruction should be inhibited is issued from the decoder 23, or a signal 35' indicating that execution of the instruction should be inhibited is issued from other circuit, such as an external peripheral unit or the like. As a result, an inhibited signal 37 is issued from an inhibit circuit 28, in response to the signal 35 or 35'. The inhibit signal 37 controls the control circuit 24 so that

the control signals 33 on the basis of the decoded instruction may not be transferred from the control circuit 24 to the execution circuit 25. As a result, the instruction is nullified, and the execution circuit 25 does not execute the operation defined by the instruction. During this time period, the execution circuit 25 either holds the previous condition or is reset.

In addition, at this moment when it has been detected that the instruction is not to be executed, the decoder 23 applies information for advancing an address of the address register 26 to an address control circuit 27. As this information 34, data representing how many bytes are included in the instruction to be nullified, could be used. This data is normally set in a first byte of the instruction which should be not executed, and therefore, the first byte of the instruction has to be read out of the memory 21 and to be decoded. For instance, if it is a 3-byte instruction, then information representing that the present content of the address register 26 (at this moment the content being the address designating the first byte of the instruction) whould be advanced by 3, is applied to the address control circuit 27. As a result, 3 is added to the present content of the address register 26 by an adder means included in the address control circuit 27, and thus the content of the register 26 is modified to the address corresponding to a next new instruction. This modification of an address needs only a very short period, and hence it can be executed within one machine cycle in which the first byte of the instruction not to be executed is

read out and decoded. Consequently, an address designation for the remainder bytes of the instruction, which should not be executed, is not performed. Therefore, in the next subsequent machine cycle, since a new address for designating an instruction to be newly read out has been already set in the address register 26, the process can immediately proceed to the execution of the next new instruction without redundantly spending a wasteful time as is the case with the prior art apparatus.

Also, even if the instruction to be nullified is a one-byte instruction, the present invention is effective in the case where the instruction necessitates a plurality of machine cycles for execution. More particularly, the apparatus according to the present invention comprises means for modifying an address without respect to an execution of an instruction, and therefore, a processing for a next new instruction can be immediately carried out after an address modification is terminated. The time period of this modification is at most one machine cycle. In this case, a reset signal 38 is applied to the inhibit circuit 28 after the modification is terminated. Thereby, the inhibit condition of the execution circuit 25 is released, and so, a machine cycle succeeding to the machine cycle of the address modification can be effectively used as a read period for a next new instruction or an execution period for the read-out new instruction without being wasted as a non-execution period.

Now description will be made on a more detailed construction of

the address control circuit 27 with reference to Figs. 4 and 5.

Fig. 4 shows one example of the address control circuit employing an adder circuit 40, in which address advancing information 34 is applied to one input, and a present content 42 of the address register 26 is applied to the other input. The result of addition is once set in a register 41 and then transferred to the address register 26 as a new address. As a matter of course, the register 41 could be omitted and replaced by the address register 26.

Fig. 5 shows another example of the address control circuit in which an incrementer 51 is used. From a register 52 which hold the present content of the address register 26, the content is input to the incrementer 51, and it is incremented by +1 each time when a signal 53 issued from a count control circuit 50. The count control circuit 50 detects the necessary number of counts on the basis of the address advance information 34 and thereby controls the incrementer 51. In this case, a signal 53 is a signal for instruction addition, and moreover, since the addition can be executed at a high speed, it is possible to achieve the operations of addition a plurality of times within one machine cycle. In this connection it is preferable to synchronize the output timing of the signal 53 with a basic operation clock signal of the apparatus.

As described above, according to the present invention, with regard to an instruction consisting of a plurality of bytes or an instruction necessitating a plurality of machine cycles, since a processing for

nullifying the instruction can be executed within an extremely shoft period of time, the program processing speed can be greatly emhanced. It is to be noted that obviously this effect of the present invention can be expected for every type of apparatuses irrespectively of whether the apparatus is of low-speed type or of high-speed type.

In the following, description will be made on other preferred embodiments of the present invention with reference to representative examples of practical instructions and procedures for their execution.

Among the instructions used in a micro-computer (for example, 4-bit system), there is a skip instruction. This means such instruction that when a predetermined condition is fulfilled within a computer and/or external units added thereto, a processing is carried out such that an instruction stored at the memory location subsequent to the skip instruction is skipped (that is, the subsequent instruction is disregarded) and execution of the further subsequent instruction (the second instructions subsequent to the skip instruction) is commenced. For instance, it is assumed that a skip instruction reading "skip, if A-register is O" and a branch or jump instruction reading "branch or jump to address X" subsequent to the skip instruction are stored consecutively in a memory. In this case, if the A-register is 1, then the branch or jump is not effected because the branch or jump instruction is skipped, but if the A-register is not O, then since the next instruction is not skipped, the next branch or jump instruction is executed, and the processing branches or jumps to address X. Accordingly, by

arranging the above-mentioned two instructions consecutively in the above-described manner, one can construct a conditional branch or jump instruction that if the A-register is not O, then branch or jump to address X, but if it is O, then not branch or jump (execute the next instruction). Moreover, by combining various other instructions with a skip instruction as subsequent instructions, the skip instruction can achieve more widely such processing that the subsequent instructions are to be executed only when the condition designated by the skip instruction is not fulfilled, and the subsequent instructions are to be disregarded when the condition designated by the skip instruction is fulfilled. The same effect as this skip instruction can be achieved by employing the conventional conditional jump instruction and selecting the destination of jump at the second instruction subsequent to the conditional jump instruction. However, in the case of the jump instruction, it is always necessary to designate the address of the jump location. Whereas, the skip instruction has a merit that regardless of how many bytes the next instruction to be disregarded may comprise, the second instruction subsequent to the skip instruction can be used without designating the address of that instruction at all. Accordingly, the skip instruction is more excellent in view of effective utilization of a memory. The skip instructions for effecting such processing have been widely and conveniently utilized especially in a micro-computer of 4-bit system or 8-bit system or the like which has a fewer kinds of instructions. Furthermore, besides the above-described skip instruction,

a processing of disregarding a certain instruction can be effectively utilized also in the following case.

More particularly, it is effectively utilized in the case where a plurality of instructions of the same type are stored in a memory as arrayed consecutively and the apparatus has such function that after a certain instruction stored at an initially designated address has been executed, if an instruction stored at the next designated address is an instruction of the same type in definition as the initial instruction, then the processing should proceed to the further next instruction while disregarding this instruction (the second instruction) (i.e. without executing this instruction). If the above-mentioned function is provided, it can be utilized conveniently as will be described below. For instance, it is assumed now that in the beginning of a certain subroutine is arranged an instruction that a value "3" should be set in an A-register, at the second location is arranged an instruction that a value "4" should be set in the A-register, and at the third location is arranged an instruction that a value "5" should be set in the A-register, and subsequent thereto a routine for carrying out a processing by making use of the content of the A-register is arranged. When the processing enters this subroutine, if it enters at the first instruction that the value "3" should be set in the A-register, then according to this subroutine at first 3 is set in the A-register, but since the next instruction, i.e., the instruction that the value "4" should be set in the A-register is an instruction of the same type as the aforementioned initially executed instruction

that the value "3" should be set in the A-register (such instructions being defined to be of the same type), the above-mentioned next instruction is disregarded as a result of the above-referred function, and so, the processing proceeds further to the next subsequent instruction that the value "5" should be set in the A-register. However, this next subsequent instruction is disregarded because it is also of the same type as the preceding instruction, and after all, in this case, the subroutine would be executed by employing the value "3" initially set in the A-register as a parameter. On the other hand, upon initially entering this subroutine, if the processing enters at the second instruction that the value "4" should be set in the A-register, then due to a similar processing to the above-described, in this case the subroutine would be executed by employing the value "4" as a parameter. Likewise, if the processing enters at the third instruction that the value "5" should be set in the A-register, then the subroutine would be executed by employing the value 5 as a parameter. In this way, one can simply prepare a subroutine in which a processing is executed with different parameters by merely variably designating the initial entrance address. Throughout this specification, the aforementioned function that defining a group of given instruction to be of the same type, in the case where instructions of the same type appear consecutively, while the initial instruction would be executed, the subsequently appearing instruction or instructions of the same type would be disregarded, is called "stack-up function of instructions".

Let us now consider the case where upon carrying out the aforementioned skip instruction or stack-up function of instructions, the correspondingly required condition has been fulfilled (that is, for a skip instruction, the case where the condition designated by this instruction gas been fulfilled, and for a stack-up function of instructions, the case where the condition that a certain instruction is of the same type as the previous instruction has been fulfilled).

In the heretofore known microprogram control apparatus, a method was employed such that during the period when the instruction is disregarded, an inhibit circuit for inhibiting the execution is activated. Therefore, for an instruction which necessitates, for example, 6 machine cycles for execution, in the heretofore known apparatus even in the case of nullifying this instruction, it was resulted that only after 6 machine cycles have been spent similarly to regular execution of this instruction, the processing can first shift to the next instruction, and so, the known apparatus had a shortcoming that the processing time was unnecessarily prolonged.

Whereas, a microprogram control apparatus according to one preferred embodiment of the present invention as will be described later, comprises a microprogram counter for designating a memory address of a micro-instruction, a decoder for determining whether or not a decoded micro-instruction belongs to a group of micro-instructions defined to be of the same type as the micro-instruction processed immediately before the decoded micro-instruction, means

for suppressing execution of the decoded micro-instruction in the event that the micro-instruction decoded by said decoder has been determined to be of the same type as the micro-instruction processed immediately before the decoded micro-instruction, and means for setting the content of said microprogram counter so as to designate the address of the micro-instruction next to the decoded microprogram in the aforementioned event.

Now the above-outlined microprogram control apparatus according to the present invention will be described in more detail with reference to Fig. 6. At first, description will be made in connection to a skip instruction. Fig. 6 is a block diagram adapted for explaining the operation in response to a skip instruction. Reference numeral 61 designates a micro-instruction memory circuit. In the case of executing a given instruction, during a first machine cycle for that instruction, at first a first byte of this instruction stored at an address designated by an address counter 63 is read out of the memory 61. Then, it is stored in an instruction registor 62. When the first byte of the micro-instruction has been read in the instruction register 62 in the above-described manner, this is decoded by a decoder 64, and depending upon whether this instruction is a 1-byte instruction, 2-byte instruction, 3-byte instruction, or else, a control circuit 65 controls a program counter control circuit 66. As a result, for each one of the machine cycles necessitated for further decoding this instruction, the content of the program counter 63 is stepped one by one, and thereby

all the bytes belonging to that instruction are successively fetched in the instruction register 62 to be decoded. In this way, the processing designated by this decoded micro-instruction is executed by an execution circuit (not shown) that is controlled by the control circuit 65. Then, after a number of machine cycles which are necessitated for executing this instruction have elapsed, the control circuit 65 further advances the content of the program counter 63 by 1 via the program counter control circuit 66. And in the next machine cycle, the first byte of the next instruction stored at the address designated by the content of this program counter 63 is read out, and thus the processing proceeds to execution of the next new instruction. Here it is assumed that a skip instruction has been fetched in the instruction register 62. As described above this is decoded by the instruction decoder 64, and due to the designation that this instruction is a skip instruction, a logic level on a skip instruction designating line 67 is turned to "1". Furthermore, the portion designating a skip condition in this skip instruction is fed to a skip condition selection circuit 68, in which among various skip conditional signals 69, a conditional signal designated by this skip condition is selected. In these conditional signals 69 are included various conditional signals such as, for example, a signal adapted to take a logic level of "1" if the content of the aforementioned A-register is O, but take a logic level of "O" if the content of the A-register is not O, a signal adapted to take a logic level of "1" if input data are present at a designated into port but take a logic of "O"

if input data are not present, or a signal indicating a detected condition of a carry output and/or a flag of an ALU. The conditional signal selected in the skip condition selection circuit 68 is combined with the signal on the above-mentioned skip instruction designating line 67 by means of an AND circuit 70, and applied to a set terminal of a flip-flop 71.

Accordingly, the flip-flop 71 can be set provided that a skip instruction has been decoded and also that the condition designated by the skip instruction is fulfilled. When the above-described processing of the skip instruction has been terminated, the control circuit 65 controls the program counter control circuit 66 so as to advance the content of the program counter 63 by 1. And in the next machine cycle, the first byte of the instruction starting from the address next to this skip instruction is fetched in the instruction register 62. In this way, since information, of how many bytes a given instruction is formed, can be obtained by decoding the first byte of instruction, this information indicating a number of bytes of an instruction is entracted and applied to one input of an adder 73 via a byte number output line 72. To the other input of this adder 73 is applied the content of the program counter 63. Now, at the time point when the first byte of the instruction was read, the decoding of the number of bytes in that instruction was completed and also the addition of the input of the byte number and the content of the program counter 63 by means of the adder 73 has been completed in the above-described

manner, the control circuit 65 outputs a program counter write pulse through a program counter write pulse line 74. This program counter write pulse line 74 and the output line of the above-mentioned flip-flop 71 are combined in logic by a second AND circuit 75. Hence, so long as the flip-flop 71 is set (that is, only when the preceding instruction was a skip instruction and the designated skip condition is fulfilled), on the output side of the second AND circuit 75 appears the write pulse issued from the control circuit 75 via the program counter write pulse output line 74. This write pulse controls the program counter control circuit 66 to write the output of the adder 73 in the program counter 63. Then, the content of this program counter 63 would designate the first byte of the second instruction subsequent to the skip instruction. The above-described processing is completed generally in the machine cycle in which the first byte of the instruction immediately subsequent to the aforementioned skip instruction. Now, the first byte of the thus designated second instruction is fetched in the instruction register 62 during the next machine cycle, and at this moment the processing first enters the first machine cycle of this instruction.

It is to be noted that control circuit 65 generates a reset pulse at a time point near to the end of this machine cycle after it generated the write pulse on the program counter write pulse line 74. This reset pulse is applied to a reset terminal of the flip-flop 71 via a reset pulse line 76, and so, the flip-flop 71 is reset. Moreover, the output of the flip-flop 71 is, in addition to serving as one input of the second AND

circuit 75, applied via an instruction execution suppress signal line 77 to the control circuit 65 as an instruction execution suppress signal. During the period when the logic level of this signal line is "1", except for the generation of the program counter write pulse and generation of the reset pulse, the other processing to be executed as a result of decoding of the first byte, is suppressed. At the same time, the trailing edge portion of this suppress signal is utilized to set the control circuit 65 so that in the next machine cycle the byte fetched in the instruction register 62 may be processed as the first byte of the next new instruction.

As a matter of course, even if a skip instruction is decoded, in the event that the condition designated by the skip instruction is not fulfilled, since the flip-flop 71 is not set, the instruction next to the skip instruction is executed according to the regular process as described in the initial part of the description of the apparatus shown in Fig. 6.

In this way, the previously described function in response to the skip instruction can be realized. As will be apparent from the above description, in the case where a skip condition designated by a skip instruction is fulfilled, regardless of the instruction subsequent to the skip instruction (regardless of how many machine cycles are necessitated for regularly executing this instruction), it is always possible to nullify that instruction and proceed to the second new instruction within at most one machine cycle.

The timing chart of the above-described processing is illustrated in Fig. 7. Fig. 7A represents the timing relation in the case of the prior art apparatus. Assuming that the memory address where an skip instruction is stored in n, initially when the content of the program counter is n , the skip instruction is read out. Here it is assumed that the skip condition is not fulfilled. Then, in the next machine cycle, the content of the program counter is incremented to n+1, and the first byte of the next instruction is read out. It is assumed that this instruction is a 3-byte instruction (for instance, a call instruction) which necessitates 6 machine cycles for its execution. In this assumed case, the content of the program counter steps up to n+3 in the subsequent two machine cycles to read out all the instruction of 3 bytes, and execution of this instruction is completed by spending further 3 machine cycles. And, in the next subsequent machine cycle, the program counter steps to n+4 to read out the first byte of the second instruction after the skip instruction, and the processing newly enters execution of this instruction. The processing in the above-described case is represented by the second diagram as noted "Skip Condition not Fulfilled" in Fig. 7A.

Whereas, in the event that the skip condition is fulfilled, during all the machine cycles in which the instruction just subsequent to the skip instruction (in the above-assumed case, a call instruction) is to executed, in place of the inherent execution of the call instruction a non-operation condition is maintained, and after the 6 machine cycles

have been spent without executing any processing effectively, the processing enters execution of the second instruction subsequent to the skip instruction. This processing is represented by the third diagram as noted "Skip Condition Fulfilled" in Fig. 7A.

On the other hand, the timing chart in the case of the apparatus according to the present invention shown in Fig. 6, is illustrated in Fig. 7B. In this case also, when the skip condition is not fulfilled, the mode of processing is quite similar to that in the case of the prior art apparatus shown in Fig. 7A. However, in the event that the skip condition is fulfilled, as shown by the third diagram as noted "Skip Condition Fulfilled" in Fig. 7B, in the next one machine cycle subsequent to the skip instruction the program counter is advanced by the number of bytes of the instruction just subsequent to the skip instruction (the number of bytes of the instruction to be skipped) at a time. In the illustrated example, the program counter is advanced by 3 at a time. Hence, in the next subsequent machine cycle, the processing can enter execution of the second instruction subsequent to the skip instruction.

It is to be noted that while the address of the second instruction was directly obtained by adding the content of the program counter and the number of bytes of the instruction just subsequent to the skip instruction by means of the adder and thereby the skip processing was achieved within a minimum processing time in the preferred embodiment of the present invention illustrated in Fig. 6, instead the skip processing can be effected in the following manner.

That is, both the output of the flip-flop 71 and the output on the byte number output line 72 are directly applied to the program counter control circuit 66. In this alternative embodiment, in the event that the output of the flip-flop 71 has become logic "1", the content of the program counter 63 could be stepped at a higher cycle repetition frequency than that employed in the regular instruction processing by a factor of the number of bytes designated by the output on the byte number output line 72. By making the above-mentioned provision, the processing time in the case where the skip condition is fulfilled, can be made shorter than that of the prior art apparatus.

While the present invention has been described above in connection to nullification of the instructions to be skipped in the case of employing a skip instruction, the invention is equally applicable to the case where the above referred "stack-up function of instructions" is utilized.

A still another preferred embodiment of the present invention is illustrated in a block diagram in Fig. 8. While reference numeral 104 designates a decoder having a similar function to the instruction decoder 64 in Fig. 6, the former is further added with the following function. That is, when a first byte of a given instruction has been fetched in an instruction register 82, the decoder 104 compares the information contained in this newly read-out byte with the information contained in the first byte of the just preceding instruction which is stored in the decoder 14, to check whether or not the newly read-out

instruction is of the same type in definition as the just preceding instruction. For this checking, it is only necessary to compare the respective instructions with respect to the principal portions of their codes or with respect to their code portions indicating the type of instructions.

In this way, if the present instruction is detected to be an instruction of the same type as the just preceding instruction, then the decoder 104 outputs a pulse of logic level "1" onto a stack-up instruction detection line 107. This detection line 107 is compled to a set terminal of a flip-flop 91, which has a similar function to the flip-flop 71 in Fig. 6, to set the flip-flop 91. Once this flip-flop 91 is set, then the subsequent processing is almost similar to that described above with reference to Fig. 6, and in a program counter 83 is written an output of an adder 93 which operates to add the present content of the program counter 83 with an output on a byte number output line 92 which output represents the number of bytes of the decoded instruction. In addition, the set output of the flip-flop 91 controls a control circuit 85 via an instruction execution suppress signal line 97 so that execution of the decoded instruction may be suppressed. In this way, the above-mentioned processing is completed within the same machine cycle in which the first byte of this instruction was read out, similarly to the processing described above with reference to Fig. 6, and in the next subsequent machine cycle, the first byte of the next new instruction is read in the instruction register 82, so that the processing can immediately

enter the machine cycle for executing the instruction.

Also, as noted previously in connection to Fig. 6, instead of employing the adder 93 in Fig. 8 the output of the flip-flop 91 and the output on the byte number output line 92 could be directly applied to a program counter control circuit 86. In this modified case, when the output of the flip-flop 91 takes a logic level "1", the content of the program counter 83 could be stepped at a higher cycle repetition frequency than that employed in the regular instruction processing by a factor of the number of bytes designated by the output on the byte number output line 92.

As described above, according to the present invention there is obtained an advantage that in the case of nullifying a subsequent instruction or instructions of the same type by employing the stack-up function of instructions, a microprogram control apparatus which can process within a shorter processing time than the prior art apparatus, is provided. Furthermore, it will be obvious that in the above-described respective embodiments, if necessary, control can be made such that an instruction appearing a plurality of steps after a given instruction may be nullified.

CLAIMS

1. An information handling apparatus comprising:

memory means for storing instructions;

addressing means, coupled to said memory means, for designating an instruction to be read out of said memory means;

means for reading an instruction designated by said addressing means out of said memory means;

decoding means for decoding a read-out instruction;

control means, coupled to said decoding means, for generating at least one control signal on the basis of the results of decoding;

executing means, coupled to said control means, for executing a processing defined by said read-out instruction in response to said control signal;

means for detecting that the processing defined by a read-out instruction is to be inhibited; and

address control means, coupled to said addressing means, for preparing an address of the next instruction by advancing the address of the instruction defining the inhibited processing within a shorter time period than the time period allotted for said inhibited processing in response to said detecting means.

2. An information handling apparatus comprising:

a memory in which a plurality of information are stored;

a first circuit for designating a memory location where information

to be read out is stored;

a second circuit for reading the designated information out of said memory;

a third circuit for inhibiting a handling of the read-out information and for generating an inhibit signal;

a forth circuit for preparing in a short period of time an address of a new information after said inhibited information; and

a fifth circuit for handling said new information which is read out of said memory by the prepared address.

3. An information handling apparatus comprising, a memory for storing instructions, an addressing circuit for designating a memory location in which an instruction is stored, a reading-out circuit for reading out the designated instruction, a decoder circuit for decoding said designated instruction and for producing decoded data, a detecting circuit for detecting whether a condition of the apparatus is a predetermined condition or not according to said decoded data and for generating a signal when the condition of the apparatus is the predetermined condition, and an address control circuit for producing an address of a next instruction following said designated instruction within one machine cycle.

4. An information processing apparatus comprising, a memory for storing instructions, a program counter for designating an address of said memory, a decoder for determining whether or not

a decoded instruction read out of said memory by said program counter belongs to a group of instructions of the same type as the instruction preceding said decoded instruction, means for suppressing execution of said decoded instruction in the event that said decoded instruction is determined to be of the same type as the instruction preceding said decoded instruction, and means for setting a next address of said decoded instruction in a short time period into said program counter.

5. An information handling apparatus comprising, a memory for storing instructions, at least one instruction being formed of a plurality of bytes, an addressing means for reading said instructions out of said memory, means for storing the number of bytes of said at least one instruction, and address control means for producing an address by means of adding the present address to said number of bytes of said at least one instruction.

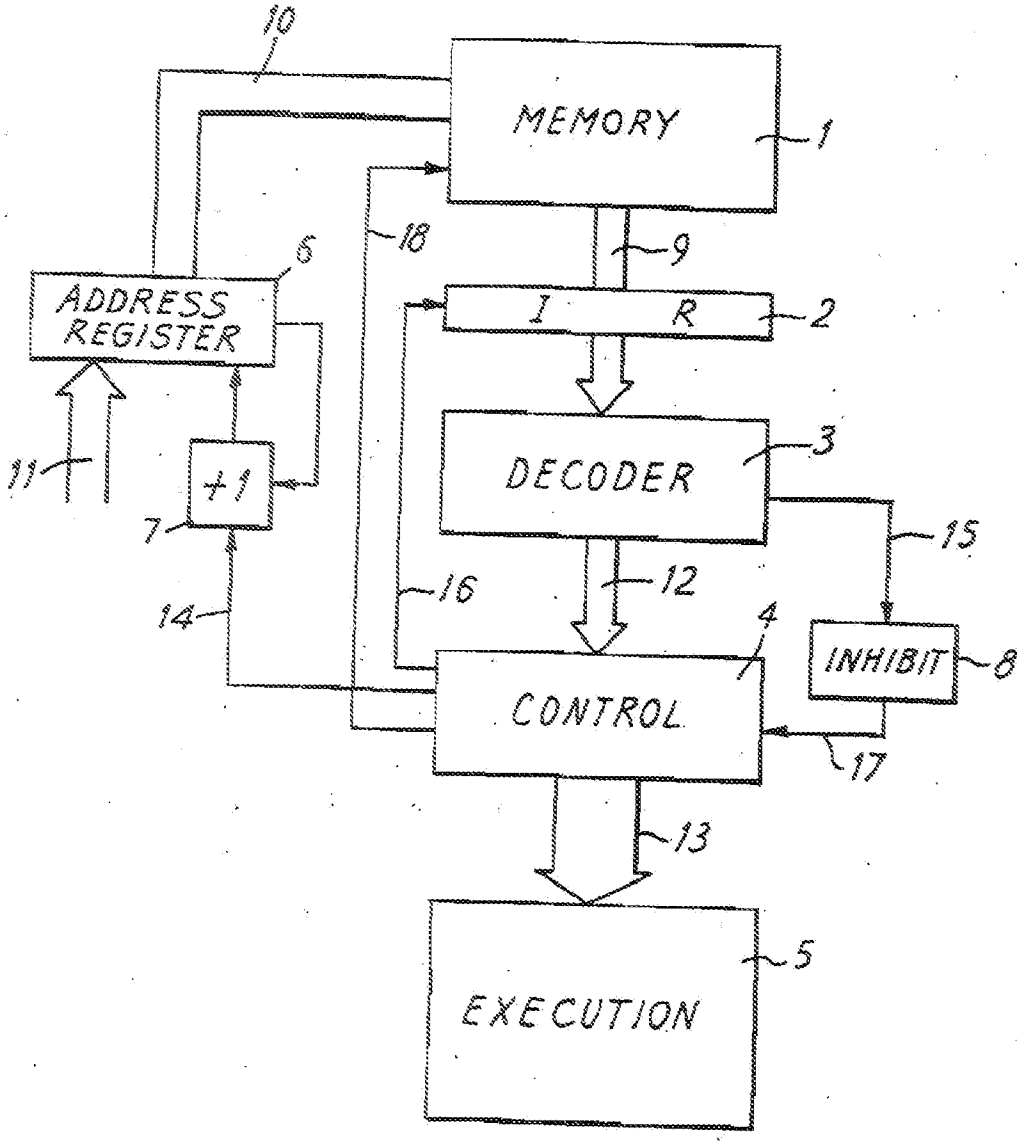



FIG.1 (PRIOR ART)

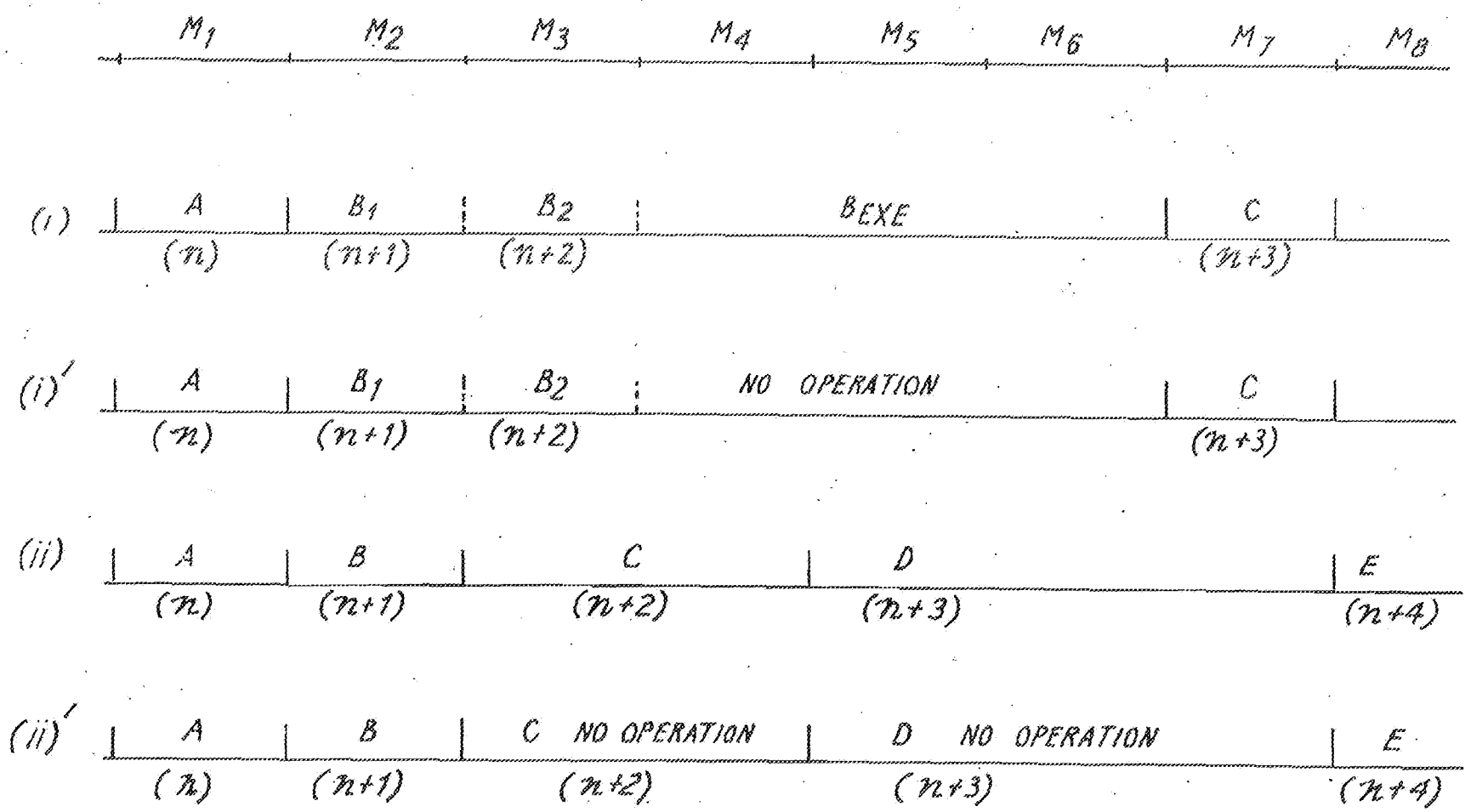



FIG. 2 (PRIOR ART)

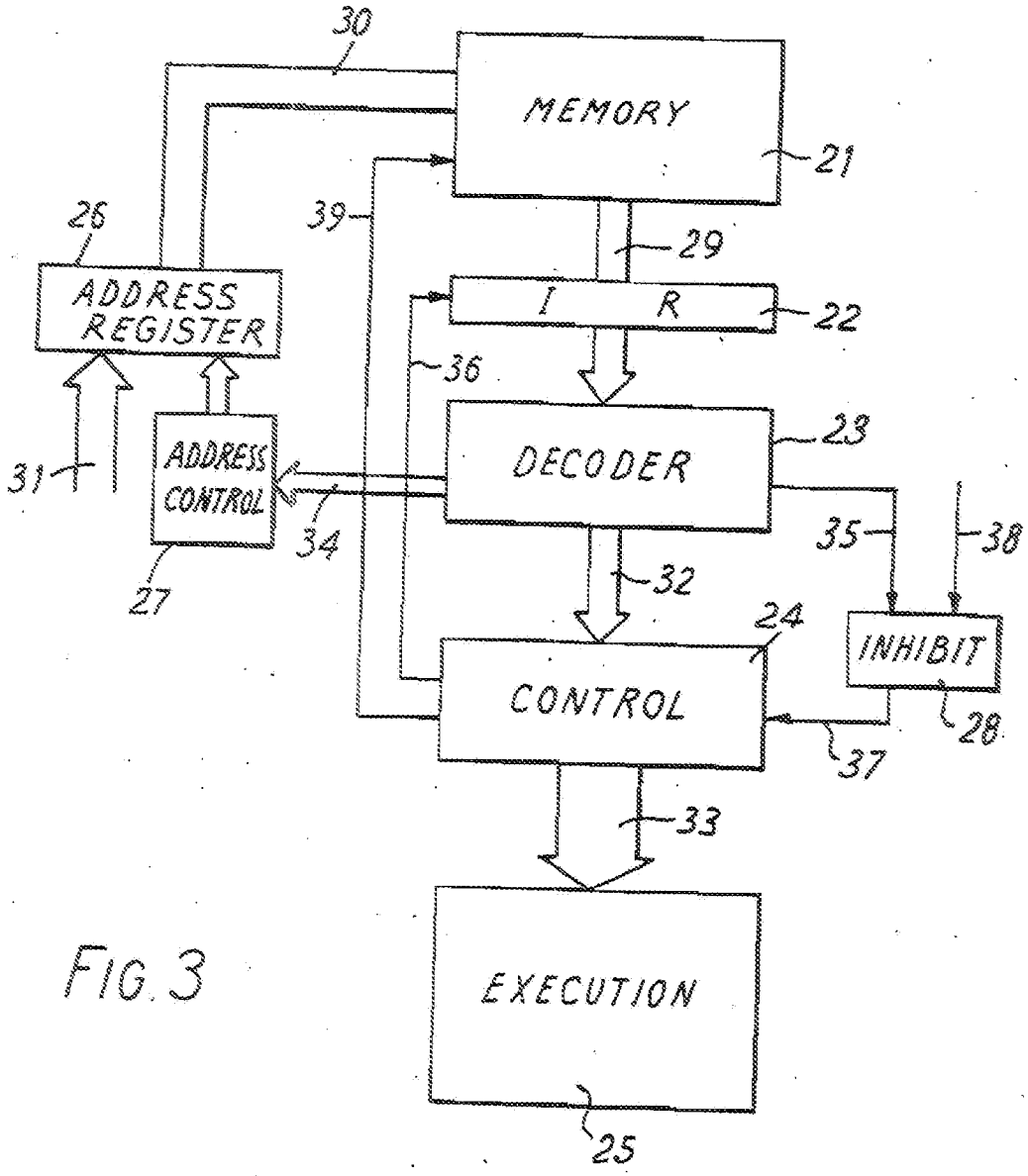

30
MEMORY
21
26
39
29
ADDRESS REGISTER
I R
22
36
23
DECODER
31
ADDRESS CONTROL
34
35
38
27
32
24
INHIBIT
CONTROL
28
37
33
EXECUTION
25

FIG. 3

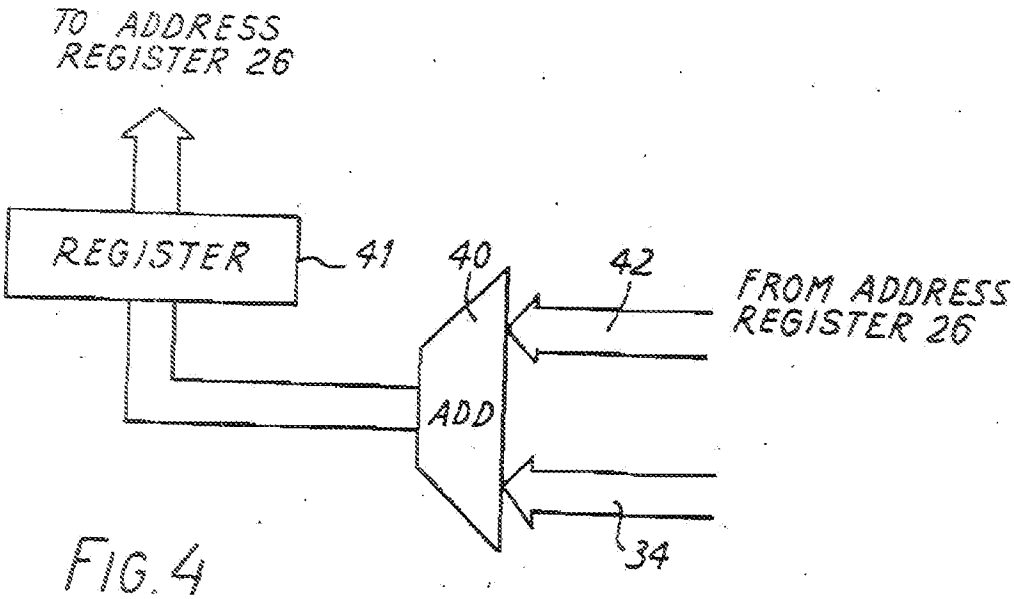

TO ADDRESS REGISTER 26
REGISTER
41
40
42
FROM ADDRESS REGISTER 26
ADD
34

FIG. 4

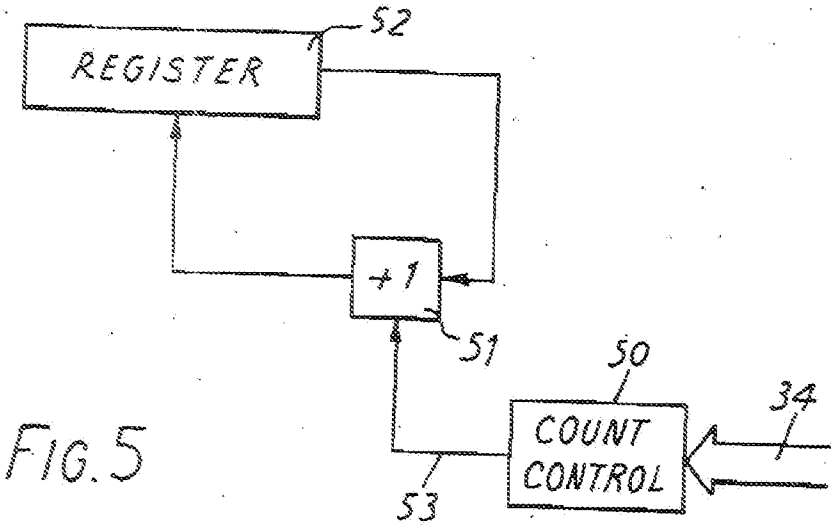

52
REGISTER
+1
51
50
34
COUNT CONTROL
53

FIG. 5

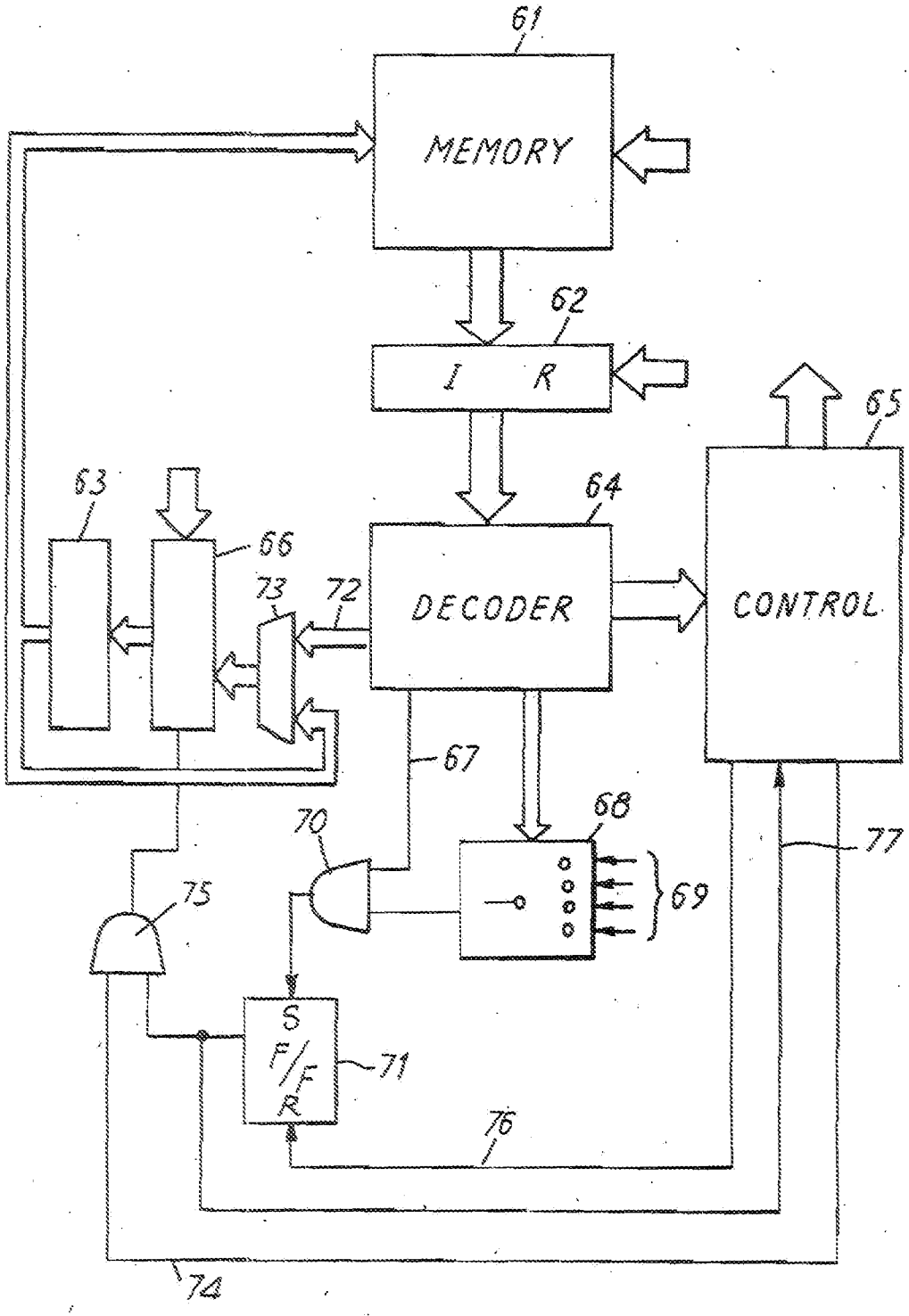

61
MEMORY
62
I R
65
63
64
66
73
72
DECODER
CONTROL
67
68
70
77
69
75
S
F/F
R
71
76
74

FIG.6

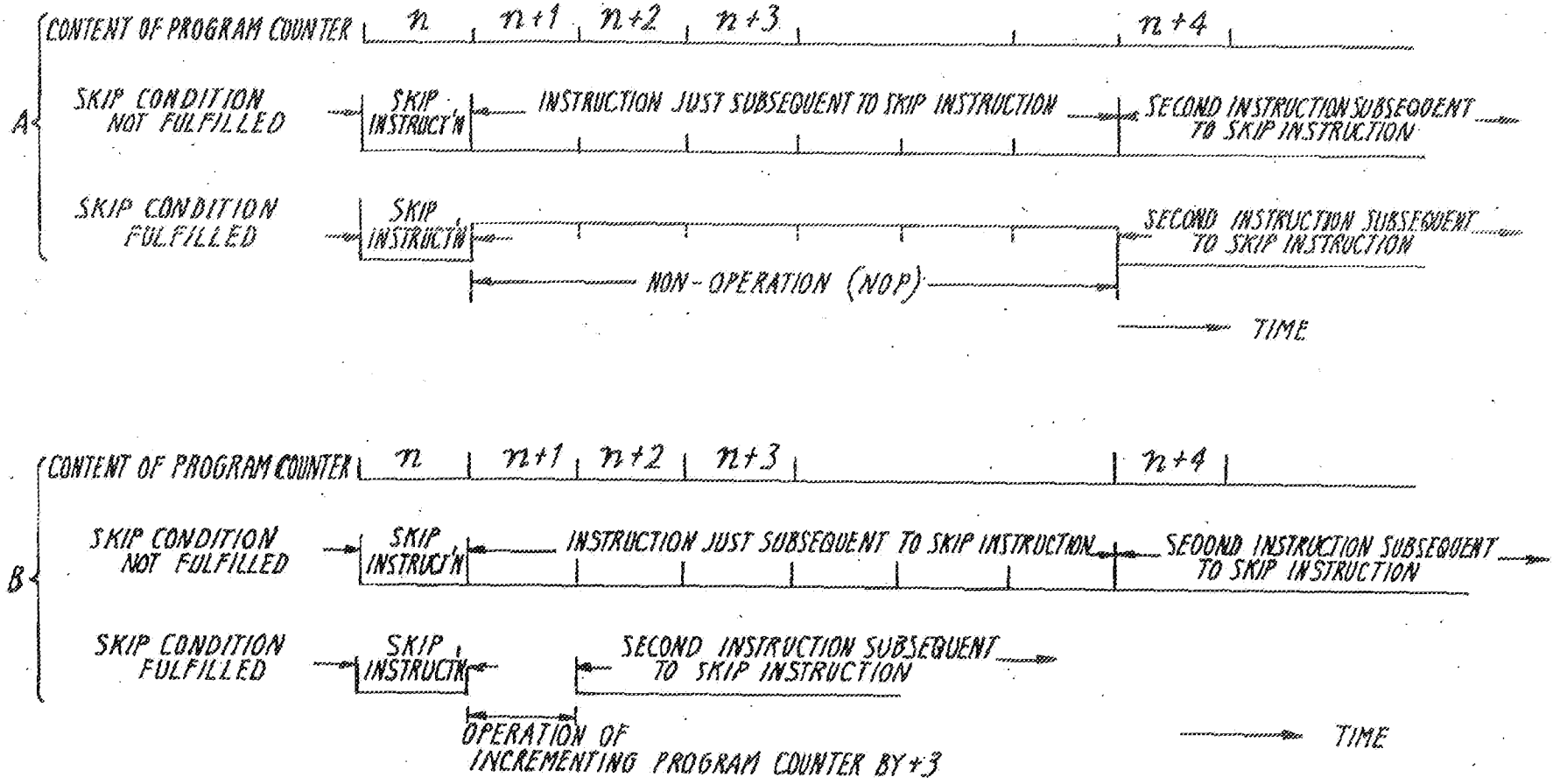



FIG. 7

6/9

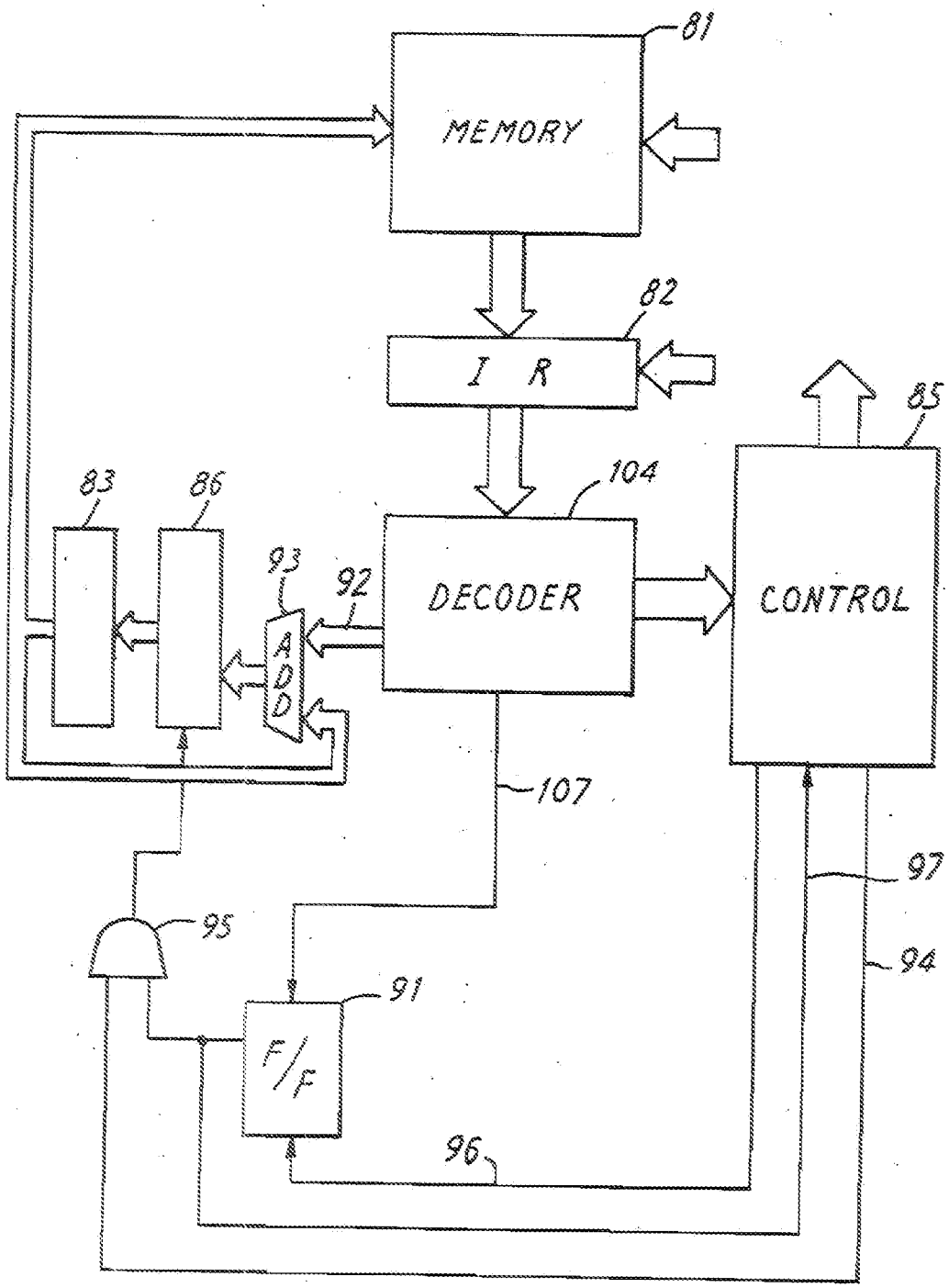

81
MEMORY
82
I R
85
83
86
104
93
92
DECODER
CONTROL
ADD
107
97
95
94
91
F/F
96

FIG. 8

European Patent Office

# EUROPEAN SEARCH REPORT

Application number

EP 81 30 5553

## DOCUMENTS CONSIDERED TO BE RELEVANT

| Category | Citation of document with indication, where appropriate, of relevant passages | Relevant to claim | CLASSIFICATION OF THE APPLICATION (Int. Cl. ³) |
|---|---|---|---|
| X | IBM TECHNICAL DISCLOSURE BULLETIN, vol. 22, no. 1, June 1979, pages 304-306, New York (USA); R.L.HOFFMAN et al.: "Branch instruction execution sequence". *Page 304, last paragraph - page 305, first paragraph* | 1-3,5 | G 06 F 9/32 |
| X | US-A-3 408 630 (BURROUGHS) *Column 4, lines 18-36; column 6, lines 21-36; column 7, lines 32-69* | 1-3,5 | |
| X | PATENTS ABSTRACTS OF JAPAN, vol. 5, no. 115, 24th July 1981, pages(P-72)(787); & JP - A - 56 57 147 (OKI DENKI KOGYO K.K.) (19-05-1981) *Abstract* | 1-3,5 | |
| A | US-A-3 577 190 (IBM) *Column 6, line 63 - column 7, line 35; column 14, lines 8-61* | 1-4 | TECHNICAL FIELDS SEARCHED (Int. Cl. ³): G 06 F 9/32, G 06 F 9/30 |

The present search report has been drawn up for all claims

| Place of search | Date of completion of the search | Examiner |
|---|---|---|
| THE HAGUE | 23-06-1982 | THOMAES K. |

CATEGORY OF CITED DOCUMENTS

X : particularly relevant if taken alone
Y : particularly relevant if combined with another document of the same category
A : technological background
O : non-written disclosure
P : intermediate document

T : theory or principle underlying the invention
E : earlier patent document, but published on, or after the filing date
D : document cited in the application
L : document cited for other reasons

& : member of the same patent family, corresponding document

EPO Form 1503. 03.82